REMARKS/ARGUMENTS

Reconsideration of this application is respectfully requested.

The rejection of claims 1-8, 10-26, 30-41, 43-61, 65-70 and 79-83 under 35 U.S.C. §103 as allegedly being made "obvious" based on Wilson '232 taken alone is respectfully traversed. Similarly, the rejection of claims 9, 27-29, 42 and 62-64 under 35 U.S.C. §103 as allegedly being made "obvious" based on a combination of Wilson '232 in view of Feistel '055 is also respectfully traversed. Furthermore, the rejection of claims 71-78 under 35 U.S.C. §103 as allegedly being made "obvious" based solely on Feistel '055 is also respectfully traversed.

Neither Wilson nor Feistel discloses the claimed features of the present invention, and they cannot be taken individually or together to provide the claimed invention(s).

Encipher claim 1 (and the decipher claim 36 and their corresponding method claims 20 and 55) requires functional modules sequentially coupled to operate sequentially on data blocks. In other words, a data block is applied to the input of functional module 1 to produce a once-modified block at the output of functional module 1. This once-modified block is applied to the input of functional module 2 to produce a twice-modified block at the output of functional module 2. This twice-modified block is applied to the input of functional module 3 to produce a thrice-modified block at the output of functional module 3, and so on. This feature is not seen in either Wilson or Feistel.

Furthermore, each functional module is claimed so as to include a data processing unit having a parallel input and parallel output and to perform a reversible process on bits applied to

the input, and, a configuring means which selects a set of the bits of a data block to apply to the input of the data processing unit, to produce at the data processing output, a replacement set of bits for the originally selected bits.

It may assist in understanding such differences to consider details of exemplary embodiments.

In the description of an exemplary embodiment of the present invention, cipher units 40 constitute functional modules, and the Fredkin's gate, as shown in Figure 5, constitutes a data processing unit. This preferred example of a data processing unit has three inputs A, B and C, and three outputs A', B' and C'. The block size for encoding/decoding is 8 bits. The configuring means generates a cipher unit design description for each cipher unit, and an exemplary design description for a particular cipher unit is given as the four segment description "010 111 110 110".

The first three segments define selection of three of the eight bits of the block and the order that they are to be applied to inputs A, B and C of the data processing unit of that cipher unit. The eight bit block is applied to the eight pins of the input of the cipher unit, and pin 3 (defined by the first segment "010") is coupled to input A, pin 8 (defined by the second segment "111") is coupled to input B, and pin 7 (defined by the first segment "110") is coupled to input C.

The bit applied to input A controls the exchange of data between inputs B, C and outputs B', C'. If the bit applied to input A is a one, then output B' is equal to input B, and similarly output C' is equal to input C. However, if the bit applied to input A is a zero,

then output B' is equal to input C, and similarly output C' is equal to input B. The output A' is always equal to input A.

The fourth segment defines whether or not outputs of the data processing unit are passed through inverters before being applied to the corresponding output pins of the cipher unit as replacement bits of the encoded block. In this example, the fourth segment is "110", which means that output A' is passed through an inverter and applied to pin 3 of the cipher unit output, output B' is passed through an inverter and applied to pin 8 of the cipher unit output, and output C' is not passed through an inverter but is applied directly to pin 7 of the cipher unit output.

Thus, it can be seen that bits at input pins 1, 2, 4, 5 and 6 of first cipher unit 40 are transferred directly to corresponding output pins 1, 2, 4, 5 and 6, while the bits at input pins 3, 7 and 8 are processed by the data processing unit to provide replacement bits at output pins 3, 7 and 8. This is not to say that the output bits are the opposite of these input bits, but that they have undergone an encoding process and their output values will depend upon that process.

The above-described process occurs at the second cipher unit in accordance with its respective design description and with the output block of the first cipher unit applied to the input pins of that second cipher unit, and then in the same fashion for each of the following cipher units of the sequential series of cipher units.

In Wilson, 64 shift registers S (each of 64 stages) form a matrix 10, and are loaded with a key having 4096 bits. This key is clocked into the registers in a preparatory phase

in which, for convenience, the registers are coupled in sequence and the key is applied to the input of register 1. Once the key has been entered into the registers, there is no further clocking of the registers.

The Wilson matrix can be thought of as having 64 inputs (one for each of the sets of AND gates OA0 to OA63), and 64 outputs (the 64 stages of the output shift register 32 OS0 to OS63).

A block of input data (64 bits) is clocked into the input shift register 18. Upon an encode command from 26, the bit in the first stage of the input register is applied to the control inputs of the first set of AND gates OA, the bit in the second stage of the input register is applied to the control inputs of the second set of AND gates OA, and so on.

The first stage of the output shift register 32 is the result of the operation of a respective exclusive OR gate 30 which has 64 inputs receiving the corresponding outputs of AND gates OA connected to the respective first stages of the 64 shift registers S of the matrix, and correspondingly for the other stages of the output shift register 32. The 64 stages of the output shift register 32 are thus loaded simultaneously during that encode command, and the encrypted block will be clocked out of that register as a new input block is clocked into the input shift register 18.

Thus, it can be seen that Wilson operates on a different principle. If it is considered *arguendo* that his matrix 10 and his exclusive OR gates 30 together constitute a functional module of the present invention, it will be seen that he does not employ sequentially coupled functional modules. Nor is there any sense in which there is a

selection of a set of the input bits to apply to a data processing unit of the hypothetical Wilson "functional module", leaving the unselected bits to be transferred directly to corresponding outputs of the functional module. The 4096 bit key contained within the 64 registers S do not select input bits for processing and permit unselected bits to be directly transferred to their corresponding outputs. The key in Wilson's "functional module" processes every bit of the input block.

Thus, it can be seen that in Wilson, the first output bit (i.e. stage OS0 of the output shift register 32) is the exclusive OR of the 64 first stage AND gate outputs X0, which themselves are the respective ANDing of the bit of the first stage of the first shift register S (i.e. the 4096th bit of the key) with the first bit of the input block, the bit of the first stage of the second shift register S (i.e. the 4032nd bit of the key) with the second bit of the input block, the bit of the first stage of the third shift register S (i.e. the 3968th bit of the key) with the third bit of the input block, and so on down to the bit of the first stage of the last shift register S (i.e. the 64th bit of the key) with the last bit of the input block.

The skilled person would not think of modifying Wilson to select a set of the bits of the input block for processing to provide "replacement" output bits, and to cascade such a stage with respective selections of bits for each of those cascaded stages.

In Feistel (and his block encipher disclosure), the user provides two 64 bit keys which are loaded into respective key-1 and key-2 shift registers 11 and 8. For a sequence of 64 clock pulses, a 64 bit block of message is fed serially with key-2 to modulo-2 adder 9, passed through multiplexer 10 and into the input of the main shift register MSR (left

hand end as seen in Figure 2). During this sequence of 64 clock pulses, a previously enciphered 64 bit block in the MSR is concurrently unloaded at its output (right hand end) while new data (the adder 9 output) is loaded at its input. This is now followed by a second sequence of 64 clock pulses during which the output of the MSR is looped back to the MSR input via input (3) of the multiplexer 10, and this new data is processed under the control of key-1 (which is looped around its register 11) by substitution devices SF-0 and SF-1, multiplexers 15 and 19 and modulo-2 adders 16 and 17.

This processing is as follows. If the key-1 bit (P) provided on the control line is a zero, then the content of field A (the leftmost 4 bits of MSR) is modulo-2 added by adder 17 with the contents of field D (the rightmost 4 bits of MSR) and the output of adder 17 becomes the new content of field D. Also, the content of field B (the next leftmost 4 bits of MSR) is modulo-2 added by adder 16 with the contents of field C (the next rightmost 4 bits of MSR) and the output of adder 16 becomes the new content of field C. If, however, the key-1 bit (P) provided on the control line is a one, then the content of field A (the leftmost 4 bits of MSR) is modulo-2 added by adder 16 with the contents of field C (the next rightmost 4 bits of MSR) and the output of adder 16 becomes the new content of field C. Also, the content of field B (the next leftmost 4 bits of MSR) is modulo-2 added by adder 17 with the contents of field D (the rightmost 4 bits of MSR) and the output of adder 17 becomes the new content of field D.

As mentioned, in the applicant's invention, the configuring means selects the predetermined set of bits at the input of a module and these are processed by the data

processing unit to provide replacement bits in the output of that module. Input bits that are not within the selected predetermined set are coupled directly to the output. Such a feature does not exist in Feistel. If the nth bit of an unloaded block from the MSR happens to have the same logic value as the nth bit of the corresponding input block, this is not because that bit was not selected for processing. Each individual bit of an input block will have been modulo-2 added with the corresponding bit of key-2, and will have been part of four fields A, four fields B, four fields C and four fields D, and the respective modulo-2 additions under the control of respective bits of key-1.

As described in Figure 4 (blocks 12 to 15) this is just one such "second" sequence, but column 4 lines 49 to 51 indicate that further such "second" sequences can be used, but even where there is a plurality of such sequences, they just perform further enciphering of one given block of input data.

So, as in Wilson, there are no cascaded modules where a block is applied to a first module, and the output of the first module is provided as an input to a second module, and so on, and where each module has a data processing unit as required by applicant's claims.

Considering now the matter of taking Wilson together with Feistel. Feistel is based on modulo-2 adding two pairs of 4 bit fields of its 64 bit MSR (A + D = D', B + C = C' or A + C = C', B + D = D', depending upon the bit value of a corresponding bit of key-1), and re-circulates the content of his 64 bit MSR. Wilson uses 64 bit shift registers, not for any re-circulating capability, but only to enter his 4096 bit key. Once the key has been

entered, the stages are not altered and merely present a 64 by 64 bit matrix of which the columns are XORed to produce the values for the output register 32, and whole rows are gated out or in depending upon the value of the corresponding bit of the input block. Not only would the skilled person not think that Wilson could be improved by adding the Feistel concept, or that Feistel could be improved by adding the Wilson concept, but he would think that it would not be possible to envisage a suitable combination because their enciphering processes are based on different principles. For example, it would not be practicable to replace the 64 Wilson shift registers with 64 Feistel MSRs, whose respective key-2s are corresponding 64 bit segments of the Wilson 4096 bit key, because the Feistel MSR enciphers the serial bit stream applied to its input from the multiplexer 10, and produces the enciphered block at the output of the MSR, whereas the Wilson shift register has no input for data to be enciphered and does not produce any enciphered data at the output of a shift register. Another way of expressing this is to appreciate that combining two such disparate methods would violate the respective reversibility (encipher/decipher) features of the two methods.

With regard to claim 71, the Examiner refers to (a) Feistel column 3 lines 62-66 as disclosing the generator of applicant's claimed apparatus (this passage states "or a much faster but somewhat less secure stream cipher cryptogram utilizing the system as a pseudo-random number generator"); (b) the last sentence of the Abstract as disclosing the encoding means of applicant's apparatus (this passage states that in stream encipherment the function of the system becomes a pseudo-random number generator whose output is

serially combined with the raw data to form the stream enciphered cryptogram, and this is entirely consistent with the column 3 stream comment); and (c) column 5 lines 6-20 as disclosing the encoding of a pseudo-random number to produce respective "descriptions" for the plurality of sequentially coupled cipher functional modules (this passage relates to block mode, not stream mode, and, as described above, the contents of the MSR are transformed by modulo-2 adding the A, B, C, D fields in accordance with key-1).

The applicant cannot identify where in this passage a pseudo-random number is encoded to provide respective descriptions of predetermined sets of bits of a data block as received at the respective module's input. Neither of the 64 bit keys (key-1, key-2) is described as a pseudo-random number, and neither key undergoes any encoding to produce a description of a predetermined set of the bits of a data block as received at a module input.

As mentioned above, there is no disclosure of "n" cascaded modules, i.e. where the output of one module is coupled to the input of the following module, and no disclosure of a description of "n" respective predetermined sets of the bits as received at the module inputs. If the Examiner is thinking that the recirculation of the contents of the MSR a plurality of times is equivalent to that plurality of cascaded modules, then one cannot find where Feistel defines a first predetermined set of the bits of the block as originally clocked into the MSR for use in a first recirculation of the contents of the MSR, and subsequent respective predetermined sets for each subsequent recirculation. Since key-1 is 64 bits long, each recirculation uses the same sequence to define the A/D, B/C (A/C,

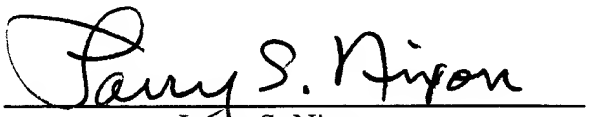
B/D) modulo-2 additions, so there is no "respective" predetermined set of the bits of the data block for each recirculation. There is no encoding of a pseudo-random number, i.e. the use of key-1 as a control input for multiplexers 15 and 19 and modulo-2 adders 16 and 17 is not equivalent to saying that key-1 is encoded, and the use of key-2 at one input of modulo-2 adder 9 is not equivalent to saying that key-2 is encoded, particularly considering that neither key is a pseudo-random number.

In view of the fundamental deficiencies of both Wilson and/or Feistel whether taken alone or in combination with respect to applicant's independent claims, it is not believed necessary at this time to point out the further deficiencies of both these references with respect to applicant's numerous dependent claims.

Accordingly, this entire application is now believed to be in allowable condition and a formal Notice to that effect is respectfully solicited.

Respectfully submitted,

NIXON & VANDERHYE P.C.

By: 

Larry S. Nixon
Reg. No. 25,640

LSN:vc
1100 North Glebe Road, 8th Floor
Arlington, VA 22201-4714
Telephone: (703) 816-4000
Facsimile: (703) 816-4100